Announcements - 2/24

- Create FP repo by tonight (midnight)
- A3 due tonight (10pm)
- FP milestone 1 due Monday
- Complete line lab by Friday night (10pm)

Announcements - 2/26

- Submit line lab by tonight
- FP milestone Monday

Announcements - 3/1

- FP milestone tonight
- Grad presentations next week
  - schedule on webpage
    - M: Caelan + Joe
    - T: Alex + Piper
    - W: Max + Ryan
  - Credit for attending: 5-minute Canvas quiz

Announcements - 3/2

- Today and tomorrow: spline lab
  - due Friday night
  - chill grading
- Milestone feedback coming soon, as needed

Lines: ✓

Modeling:

$y = mx+b$
$ax+by+c=0$ } → (implicit)

$\vec{p}+t\vec{d}$ } → (parametric)

Rendering: Bresenham/midpoint alg.

Curves:

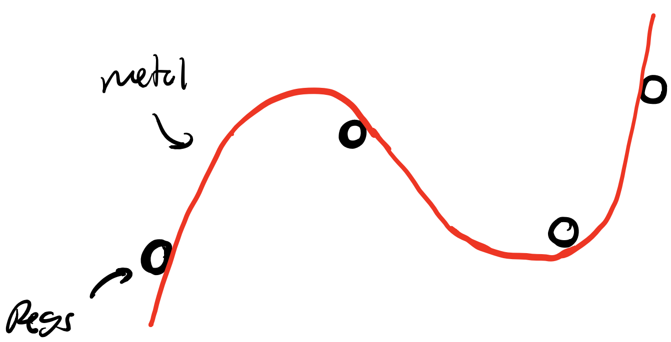
Modeling: piecewise (low-degree) polynomial pieces

Rendering: GOTO modeling
piecewise linear segments

Modeling Curves: The old-fashioned way, used in ship building

→ Control: pegs

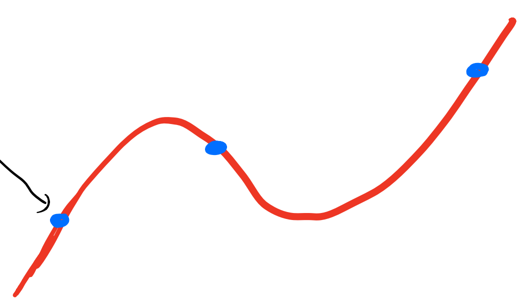
→ Smoothness: physics



Modeling Curves: The Math Way

→ control: control points

→ smoothness:
low-deg polynomial

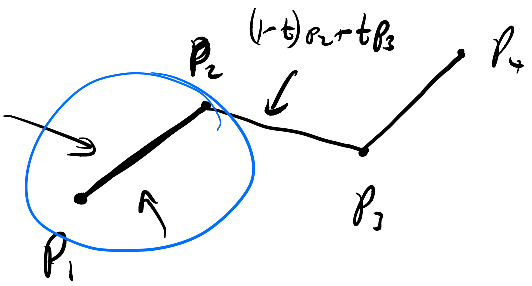


Simplest Case: degree-1 polynomials

a.k.a. line segments

t: 0 to 1

$(1-t)p_1 + t p_2$



Two problems:

(1) not smooth - need higher-degree polynomials

(2) approach doesn't generalize nicely

Reminder: linear interpolation ("lerp")

Have: $f(u) = (1-u)p_0 + u p_1$

Problem: not obviously polynomial
Solution: convert to something that is.

Notation:

- u is like t, but by convention varies only from 0 to 1.
- p_0, p_1 are points (of any dimension!)

Want:

$$f(u) = u^0 a_0 + u^1 a_1 = \begin{bmatrix} 1 & u \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \vec{u} \cdot \vec{a}$$

Task: find p in terms of a

$$\begin{array}{llcccc} & & & 1\ (u^0) & & u^1 \\ \text{start} & p_0 = f(0) & = & \boxed{1}\ a_0 & + & \boxed{0}\ a_1 \\ \text{end} & p_1 = f(1) & = & \boxed{1}\ a_0 & + & \boxed{1}\ a_1 \end{array}$$

Matrix Form:

$$\begin{bmatrix} p_0 \\ p_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$$

Control points $\rightarrow$ $\vec{p} = C \, \vec{a} \leftarrow$ coefficients

constraint matrix

To draw curves, we have p, want a.

$$\boxed{C^{-1} \vec{p} = \vec{a}}$$

Recall: given $\vec{a}$ and a parameter value u, $f(u) = \vec{u} \cdot \vec{a}$

$[1 \;\; u]$ $\quad \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}$

Let $\boxed{B = C^{-1}}$. Then:

$$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$

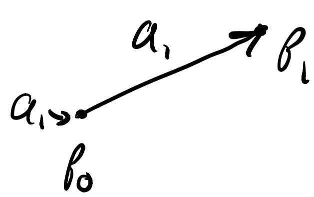
C $\quad$ B

$$f(u) = \vec{u}^T \vec{a} = \vec{u}^T B \vec{p}$$

$[1 \;\; u]$

Basis matrix $\quad$ ctrl pts

$$\begin{matrix} p_0 \\ p_1 - p_0 \end{matrix} = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} \begin{matrix} p_0 \\ p_1 \end{matrix}$$

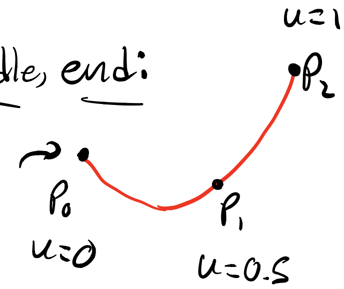


Ok but we already knew how to draw lines.

Let's apply the same machinery to quadratics!

$$f(u) = a_0 \overset{\downarrow}{u^0} + a_1 \overset{\downarrow}{u^1} + \overset{\overset{\text{new!}}{\downarrow}}{a_2 u^2} = \vec{u} \cdot \vec{a}$$

Instead, define control points: beginning, middle, end:



$$\begin{array}{llcccccccc}
 & & & u^0=1 & & u^1 & & u^2 & \\
\text{start} & P_0 = f(0) & = & \boxed{1}\, a_0 & + & \boxed{0}\, a_1 & + & \boxed{0}\, a_2 \\
\text{mid} & P_1 = f(0.5) & = & \boxed{1}\, a_0 & + & \boxed{0.5}\, a_1 & + & \boxed{0.25}\, a_2 \\
\text{end} & P_2 = f(1) & = & \boxed{1}\, a_0 & + & \boxed{1}\, a_1 & + & \boxed{1}\, a_2
\end{array}$$

$\vec{p} = C\vec{a}$

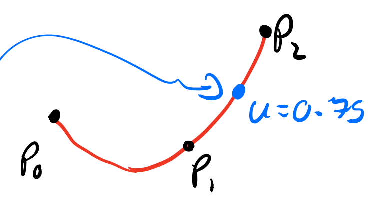
C^{-1}

$\vec{a} = B\vec{p}$

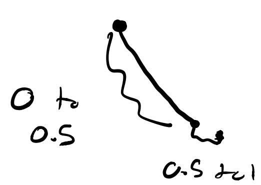
$$C = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{pmatrix} \qquad B = C^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -3 & 4 & -1 \\ 2 & -4 & 2 \end{bmatrix}$$

As before: $f(u) = \vec{u}^T B \vec{p}$

$f(0.75)\ [1 \quad 0.75 \quad 0.75^2] \cdot B\vec{p}$

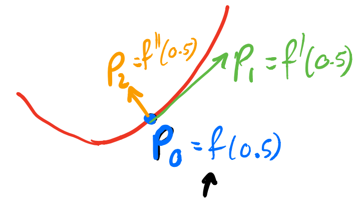


for u=0:.01:1
draw pt(f(u))



New kind of control: specify the **derivatives** at a point.

Example: Quadratic defined by:



- its middle
- its derivative at the middle
- its 2nd derivative at the middle

Pre-compute derivatives:

$$f(u) = a_0 + u a_1 + u^2 a_2$$
$$\rightarrow f'(u) = a_1 + 2u a_2$$
$$\rightarrow f''(u) = 2a_2$$

Constraints:

$$\begin{array}{lllll} & & u^0=1 & u^1 & u^2 \\ \text{mid} & P_0 = f(0.5) & = 1\,a_0 & + 0.5\,a_1 & + 0.25\,a_2 \\ \text{deriv @ mid} & P_1 = f'(0.5) & = 0\,a_0 & + 1\,a_1 & + 1\,a_2 \\ \text{2nd deriv @ mid} & P_2 = f''(0.5) & = 0\,a_0 & + 0\,a_1 & + 2\,a_2 \end{array}$$

$$C = \begin{bmatrix} 1 & 0.5 & 0.25 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{bmatrix} \quad C^{-1} = B = \begin{bmatrix} 1 & 0.5 & 0.125 \\ 0 & 1 & -0.5 \\ 0 & 0 & 0.5 \end{bmatrix}$$

Hold on: P_0 is a vector - how can you make $\vec{p}$?

(1) transpose things carefully, do mat-mat multiplication

or (2) do everything one dim at a time:

$$f(u) = \vec{u}^T B \vec{p}$$

$$\begin{pmatrix} a_{0x} \\ a_{1x} \\ a_{2x} \end{pmatrix} = \begin{bmatrix} & & \\ & B & \\ & & \end{bmatrix}_{3\times 3} \begin{pmatrix} \vec{p}_{0x} \\ \vec{p}_{1x} \\ \vec{p}_{2x} \end{pmatrix}$$

What people actually use: <u>Cubic</u> polynomials.

$$f(u) = a_0 + u a_1 + u^2 a_2 + u^3 a_3$$

Example: <u>Hermite Spline</u>: cubic defined by

- Start (1) and end (3) <u>points</u>
- Start (2) and end (4) <u>derivatives</u>

$$f(u) = a_0 + u a_1 + u^2 a_2 + u^3 a_3$$

$$f'(u) = \qquad a_1 + 2u a_2 + 3u^2 a_3$$

$$P_1 = f(0) = 1 a_0 + 0 \cdot a_1 + 0 a_2 + 0 a_3$$

$$P_2 = f'(0) = 0 a_0 + 1 a_1 + 0 a_2 + 0 a_3$$

$$P_3 = f(1) = 1 a_0 + 1 a_1 + 1 a_2 + 1 a_3$$

$$P_4 = f'(1) = 0 a_0 + 1 a_1 + 2 a_2 + 3 a_3$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -3 & -2 & 3 & -1 \\ 2 & 1 & -2 & 1 \end{bmatrix}$$

Bézier Curves

$$\boxed{\begin{aligned} f(u) &= a_0 + u a_1 + u^2 a_2 + u^3 a_3 \\ f'(u) &= a_1 + 2u a_2 + 3u^2 a_3 \end{aligned}}$$

start $\quad f(0) = \boxed{P_0 = a_0}$

end $\quad f(1) = \boxed{P_3 = a_0 + a_1 + a_2 + a_3}$

dstart $\quad f'(0) = 3(P_1 - P_0) = a_1$

dend $\quad f'(1) = 3(P_3 - P_2) = a_1 + 2a_2 + 3a_3$

$$3(P_1 - a_0) = a_1$$

$$3P_1 - 3a_0 = a_1$$

$$\boxed{P_1 = a_0 + \frac{1}{3} a_1}$$

$$3(P_3 - P_2) = a_1 + 2a_2 + 3a_3$$

$$3a_0 + 3a_1 + 3a_2 + 3a_3 - 3P_2 = \cancel{a_1} + \cancel{2a_2} + 3a_3$$

$$\frac{3a_0 + 2a_1 + a_2}{3} = P_2$$

$$\boxed{a_0 + \frac{2}{3} a_1 + \frac{1}{3} a_2 = P_2}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \frac{1}{3} & 0 & 0 \\ 1 & \frac{2}{3} & \frac{1}{3} & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} \qquad C^{-1} = B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

Bézier Basis Matrix

$f(u) = \vec{u}^T B \vec{p}$

Why is it called a <u>basis</u> matrix?

For computation: precompute $B\vec{p}$

→ poly coefficients a_i

Another interpretation: precompute $u^T B$

→ vector of <u>weights</u> applied to each p_i

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{pmatrix}$$

Example: Bézier

$$(u^T B)^T = \begin{bmatrix} 1 - 3u + 3u^2 + u^3 \\ 3u - 6u^2 + 3u^3 \\ 3u^2 - 3u^3 \\ u^3 \end{bmatrix} = \begin{bmatrix} (1-u)^3 \\ 3u(1-u)^2 \\ 3u^2(1-u) \\ u^3 \end{bmatrix} \begin{array}{l} \leftarrow P_0 \\ \leftarrow P_1 \\ \vdots \\ \end{array}$$

These are <u>Bernstein</u> polynomials

exist for <u>any</u> order

$$f(u) = (1-u)^3 P_0 + 3u(1-u)^2 P_1 + 3u^2(1-u) P_2 + u^3 P_3$$

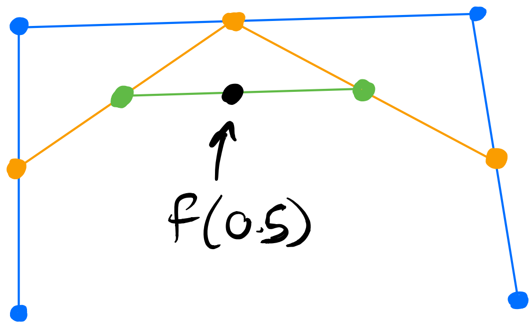
Slide: plot these

De Casteljau's Algorithm

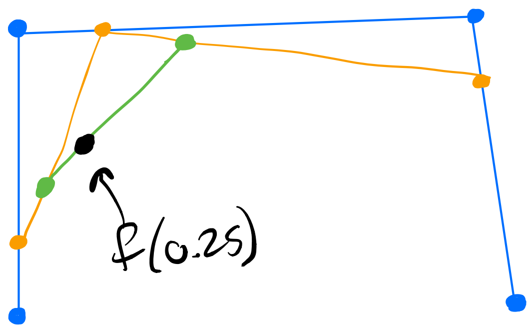
An alternative way to evaluate a point on a Bézier curve.

↳ doesn't require basis matrix! $[1 \quad 0.5 \quad 0.5^2 \quad 0.5^3]$

Geometrically: evaluate at $u = 0.5$



At $u = 0.25$:

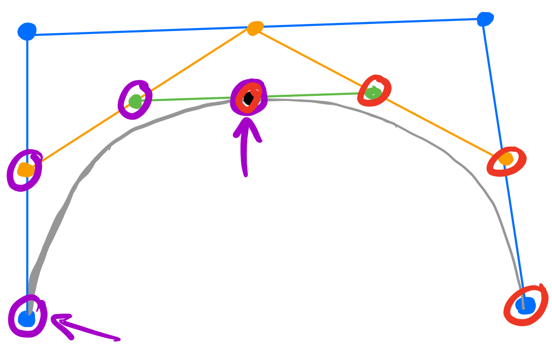


Slide: dCJ animation

Bézier Subdivision using de Casteljau's Algorithm:

Subdivide at

$u = 0.5$



$P_1^* P_2^* P_3^* P_4^*$ ctrl pts for right

$P_1' P_2' P_3' P_4'$

Control pts for
left half of curve!

half of curve!

Drawing by subdivision:

```
draw_dcj(ctrl_pts) {
    if ctrl_pts are sufficiently collinear:
        connect w/ line segments
        return
    ctrl_left, ctrl_right = subdivide(ctrl_pts, 0.5)
    draw_dcj(ctrl_left)
    draw_dcj(ctrl_right)
```